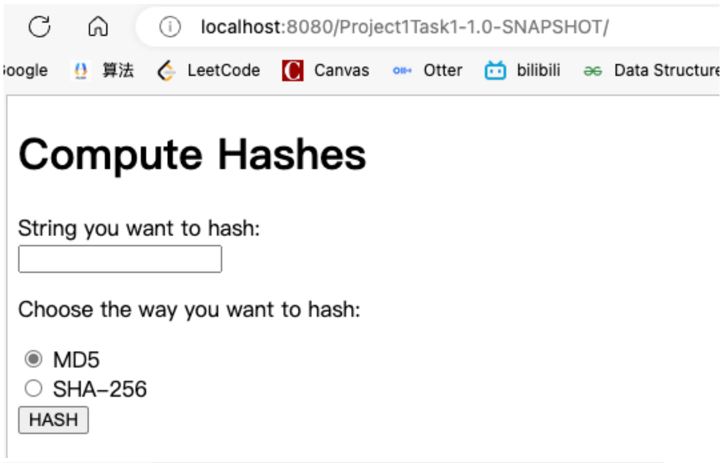
Task 1

Screen Shots

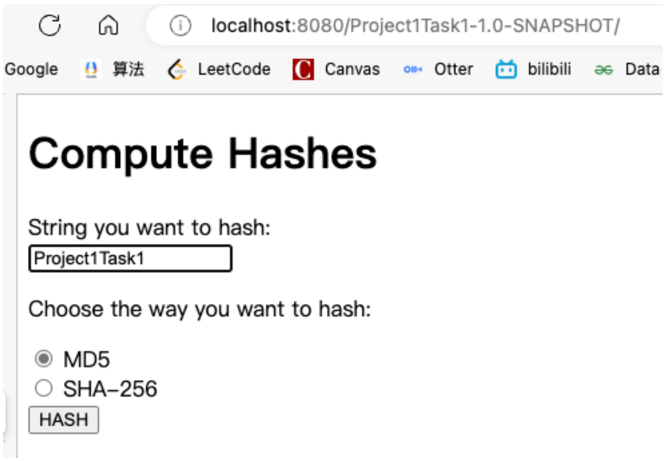


Compute Hashes

String you want to hash:

Choose the way you want to hash:

[✓ MD5]
SHA–256
HASH

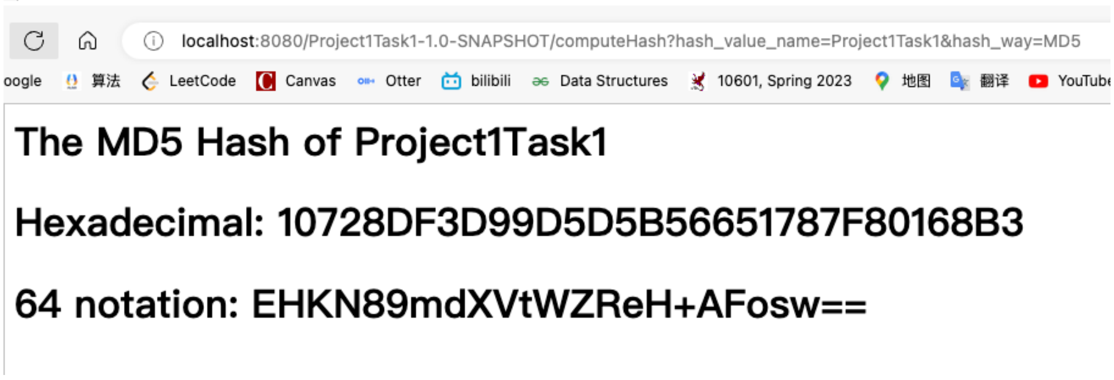


Compute Hashes

String you want to hash:
Project1Task1

Choose the way you want to hash:

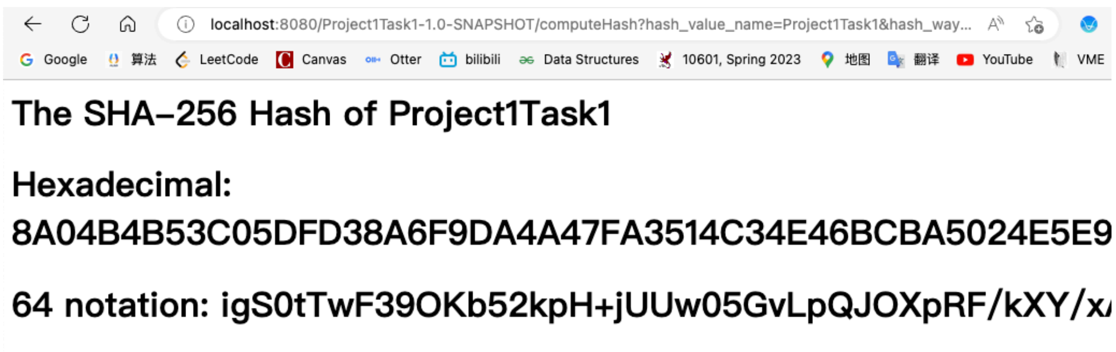
[✓ MD5]
SHA–256
HASH


localhost:8080/Project1Task1-1.0-SNAPSHOT/computeHash?hash_value_name=Project1Task1&hash_way=MD5
oogle
算法
LeetCode
Canvas
Otter
bilibili
Data Structures
10601, Spring 2023
地图
翻译
YouTube

The MD5 Hash of Project1Task1

Hexadecimal: 10728DF3D99D5D5B56651787F80168B3

64 notation: EHKN89mdXVtWZReH+AFosw==


localhost:8080/Project1Task1-1.0-SNAPSHOT/computeHash?hash_value_name=Project1Task1&hash_way...
Google
算法
LeetCode
Canvas
Otter
bilibili
Data Structures
10601, Spring 2023
地图
翻译
YouTube
VME

The SHA–256 Hash of Project1Task1

Hexadecimal:
8A04B4B53C05DFD38A6F9DA4A47FA3514C34E46BCBA5024E5E9

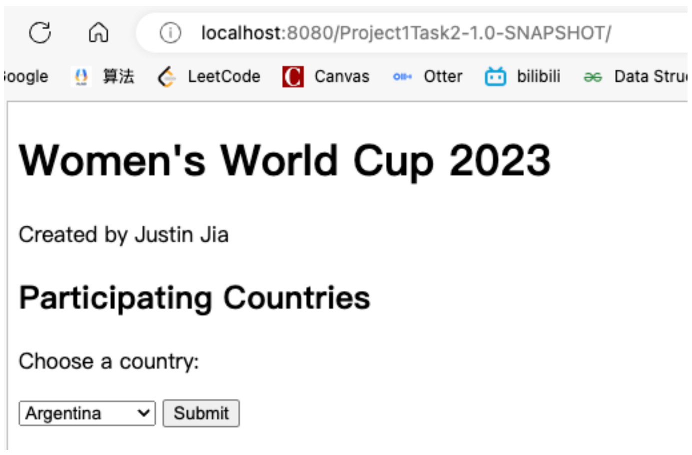
64 notation: igS0tTwF39OKb52kpH+jUUw05GvLpQJOXpRF/kXY/x

Code Snippet:

```
String hash_hexadecimal;
String hash_64notation;
MessageDigest md;

// Check which type of hash to compute
if(hash_way.equals("SHA-256")) {
    md = MessageDigest.getInstance("SHA-256");
} else {
    md = MessageDigest.getInstance("MD5");
}

// Update the message digest with the original value
md.update(original_value.getBytes());
byte[] digest = md.digest();

// Compute the hexadecimal and 64 notation representations of the
digest
hash_64notation =
jakarta.xml.bind.DatatypeConverter.printBase64Binary(digest);
hash_hexadecimal =
jakarta.xml.bind.DatatypeConverter.printHexBinary(digest);
```

Task 2

Screen Shots



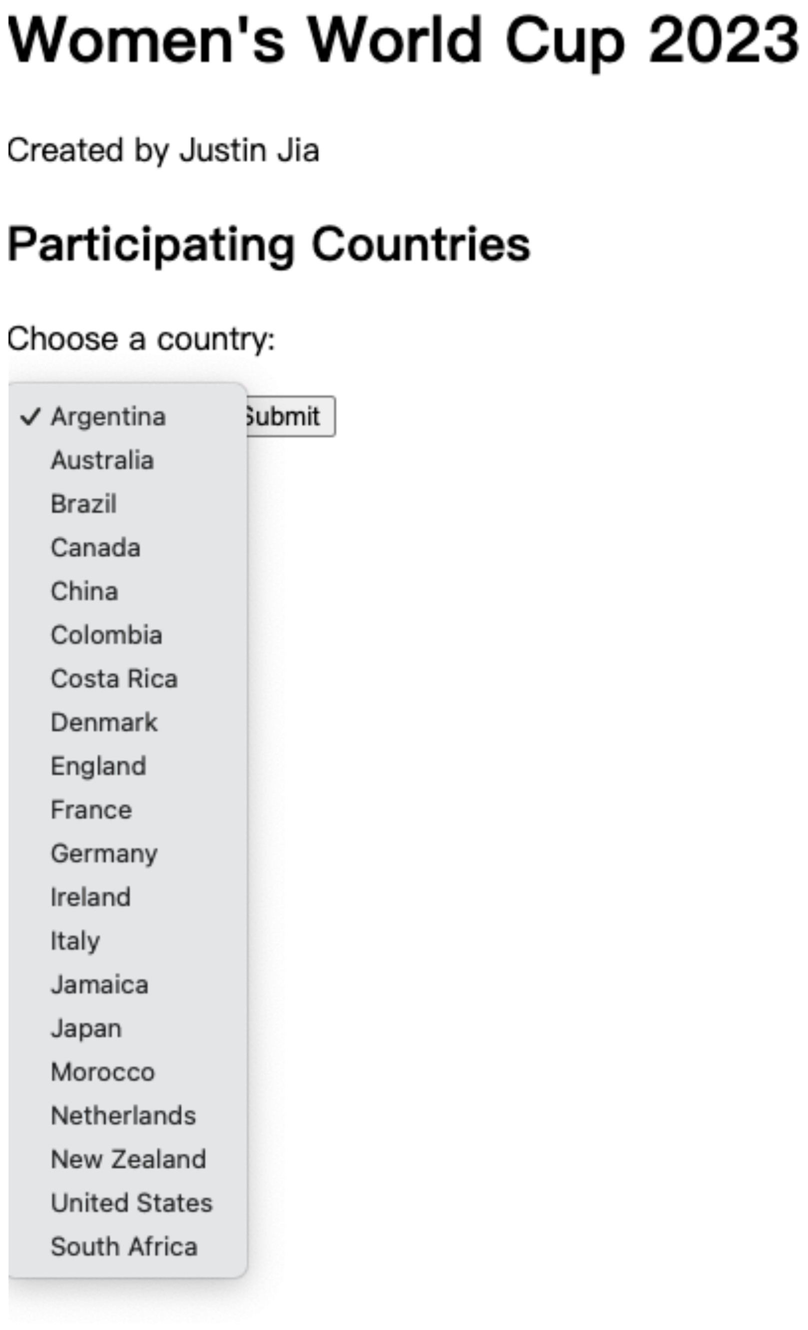
Women's World Cup 2023

Created by Justin Jia

Participating Countries

Choose a country:

Argentina Submit



Women's World Cup 2023

Created by Justin Jia

Participating Countries

Choose a country:

- ✓ Argentina
- Australia
- Brazil
- Canada
- China
- Colombia
- Costa Rica
- Denmark
- England
- France
- Germany
- Ireland
- Italy
- Jamaica
- Japan
- Morocco
- Netherlands
- New Zealand
- United States
- South Africa

Country: England

Nickname: The Lionesses

https://www.topendsports.com/sport/soccer/team-nicknames-women.htm

Capital City: London

https://restcountries.com/v3.1/name/

Top Scorer in 2019: Ellen White,6

https://www.espn.com/soccer/stats/_/league/FIFA.WWC/season/2019/view/scoring

Flag:

https://www.cia.gov/the-world-factbook/countries/

Flag Emoji:

https://cdn.jsdelivr.net/npm/country-flag-emoji-json@2.0.0/dist/index.json

Continue

Country: South Africa

Nickname: Banyana Banyana

https://www.topendsports.com/sport/soccer/team-nicknames-women.htm

Capital City:

https://restcountries.com/v3.1/name/

Top Scorer in 2019: N/A

https://www.espn.com/soccer/stats/_/league/FIFA.WWC/season/2019/view/scoring

Flag:

https://www.cia.gov/the-world-factbook/countries/

Flag Emoji:

https://cdn.jsdelivr.net/npm/country-flag-emoji-json@2.0.0/dist/index.json

Continue

Code Snippet

- api call for the flag emoji JSON record, including the conversion to a Java array of objects.

```
• private static List<FlagEmoji> flagEmojis = new ArrayList<>();

public static String searchFlagEmoji(String countryName) {
    try {
        URL url = new URL("https://cdn.jsdelivr.net/npm/country-
flag-emoji-json@2.0.0/dist/index.json");
        BufferedReader reader = new BufferedReader(new
InputStreamReader(url.openStream()));

        Gson gson = new Gson();
        Type flagEmojiListType = new
TypeToken<ArrayList<FlagEmoji>>(){}.getType();
        flagEmojis = gson.fromJson(reader, flagEmojiListType);

        String imageURL = "";
        for (FlagEmoji flagEmoji : flagEmojis) {
            if
(flagEmoji.getName().equalsIgnoreCase(countryName)) {
                imageURL = flagEmoji.getImage();
                break;
            }
        }

        if (imageURL.isEmpty()) {
            return "Flag emoji image not found for the country: "
+ countryName;
        } else {
            return imageURL;
        }
    } catch (Exception e) {
        e.printStackTrace();
        return "An error occurred while searching for flag
emoji";
    }
}
// class to represent a flag emoji
```

```java
private static class FlagEmoji {
    private String name;
    private String code;
    private String emoji;
    private String unicode;
    private String image;

    public FlagEmoji(String name, String code, String emoji,
String unicode, String image) {
        this.name = name;
        this.code = code;
        this.emoji = emoji;
        this.unicode = unicode;
        this.image = image;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getCode() {
        return code;
    }

    public void setCode(String code) {
        this.code = code;
    }

    public String getEmoji() {
        return emoji;
    }

    public void setEmoji(String emoji) {
        this.emoji = emoji;
    }

    public String getUnicode() {
        return unicode;
    }
    public void setUnicode(String unicode) {
        this.unicode = unicode;
    }

    public String getImage() {
        return image;
    }

    public void setImage(String image) {
        this.image = image;
    }
}
```

- scraping of the nickname

```
public String getNickname(String searchTag){
    String nicknameResponse =
fetch("https://www.topendsports.com/sport/soccer/team-
nicknames-women.htm");
    int cutLeft = nicknameResponse.indexOf(searchTag +
"</td><td>") + searchTag.length() + 9;
    int cutRight = nicknameResponse.indexOf("</td>", cutLeft);
    if (cutLeft == -1 || cutRight == -1) {
        return "Not found";
    }
    String nickname = nicknameResponse.substring(cutLeft,
cutRight);
    return nickname;
}
```

- scraping of the capital

```
public String getCapital(String searchTag){
    switch (searchTag) {
        case "England":
            searchTag = "United%20Kingdom";
            break;
        default:
            break;
    }
    String capitalResponse =
fetch("https://restcountries.com/v3.1/name/" + searchTag);
    int cutLeft = capitalResponse.indexOf("\"capital\":[\"");
    int cutRight =
capitalResponse.indexOf("\"],\"altSpellings\"", cutLeft);
    if (cutLeft == -1 || cutRight == -1) {
        return "";
    }
    String capital = capitalResponse.substring(cutLeft + 12,
cutRight);
    System.out.println(capital);
    return capital;
}
```

- scraping of the top scorer with number of goals

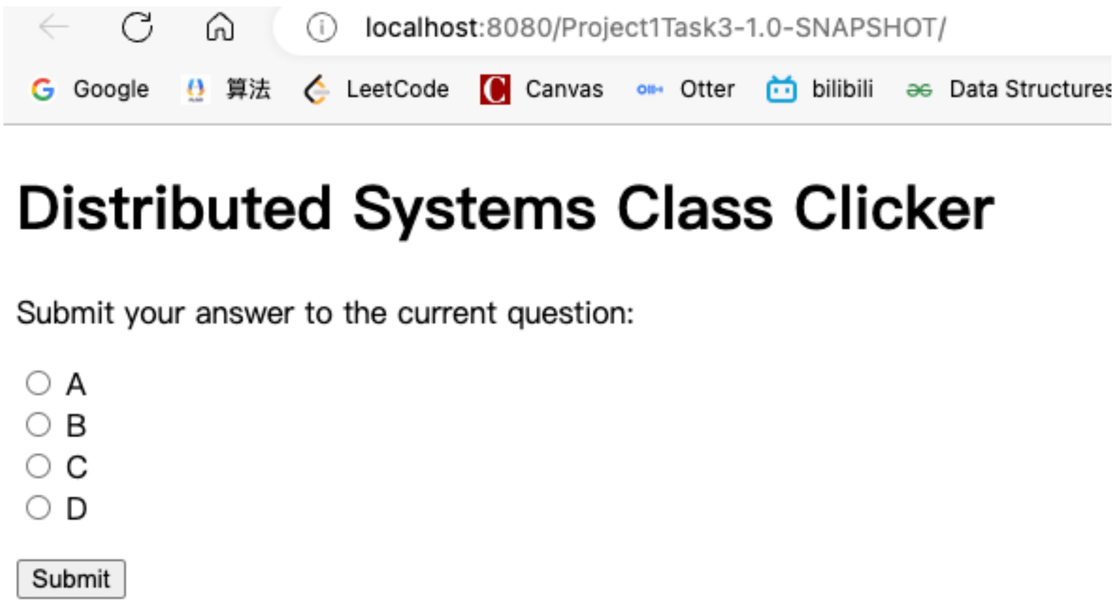
```
public String getTopScorer(String searchTag){
    String player = "N/A";
    try {
        // Fetch the top scorer response from the URL
        String topScorerResponse =
fetch("https://www.espn.com/soccer/stats/_/league/FIFA.WWC/seas
on/2019/view/scoring");

        // Find the index of the search tag in the response
        int pointIndex = topScorerResponse.indexOf(searchTag);

        // Find the end index of the player's name
        int endIndex =
topScorerResponse.lastIndexOf("</a></span></td>",pointIndex);

        // Find the start index of the player's name
        int startIndex =
topScorerResponse.lastIndexOf(">",endIndex) + 1;

        // Extract the player's name
        String playerName =
topScorerResponse.substring(startIndex, endIndex);

        // Find the start index of the player's score
        int startPointIndex =
topScorerResponse.indexOf("\"tar\">",pointIndex+97) + 1;

        // Find the end index of the player's score
        int endPointIndex =
topScorerResponse.indexOf("</span>",startPointIndex);

        // Extract the player's score
        String playerScore =
topScorerResponse.substring(startPointIndex+5, endPointIndex);

        // Combine the player's name and score into a single
string
        player = playerName + "," +playerScore;

        // Return `null` if the player string is less than or
equal to 3 characters in length
        if (player.length() <= 3 || player.length() >= 50) {
            return "N/A";
        }
        if (startIndex < 0 || endIndex >=
topScorerResponse.length() || endIndex <= startIndex) {
            return "N/A";
        }
        if (startPointIndex < 0 || endPointIndex >=
topScorerResponse.length() || endPointIndex <= startPointIndex)
{
            return "N/A";
        }
    } catch (Exception e) {
        // Handle the exception and log it
        e.printStackTrace();
    }
    // Return the combined player string
    return player;
}
```

- scraping of the flag

```
• public String flagSearch(String searchTag) throws
UnsupportedEncodingException {
    switch (searchTag) {
        case "England":
            searchTag = "United Kingdom";
            break;
        default:
            break;
    }
    searchTag = searchTag.replace(" ", "-");
    // Replace spaces in the search tag with hyphens to match
the URL format
    String searchTagLower = searchTag.toLowerCase();

    // Fetch the flag response from the URL
    String flagResponse = fetch("https://www.cia.gov/the-world-
factbook/countries/" + searchTagLower + "/flag");

    // Find the index of ".jpg" in the response
    int cut1 = flagResponse.indexOf(".jpg");

    // Find the start index of the flag URL
    int cutLeft = flagResponse.lastIndexOf("<a href=", cut1);

    // Find the end index of the flag URL
    int cutRight = flagResponse.indexOf("\" class=\"mb0\"",
cutLeft);

    // Extract the flag URL from the response
    String flagURL = "https://www.cia.gov" +
flagResponse.substring(cutLeft + 9, cutRight);

    // Print the extracted URL
    System.out.println(flagURL);

    // Return the extracted URL
    return flagURL;
}
```

Task 3

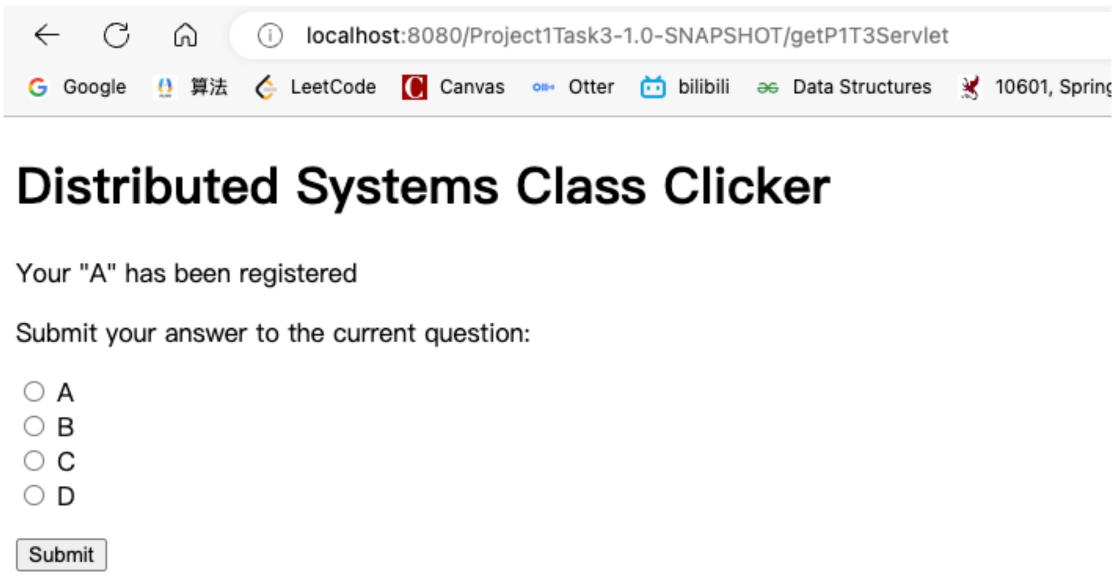
Screen Shots


localhost:8080/Project1Task3-1.0-SNAPSHOT/

Distributed Systems Class Clicker

Submit your answer to the current question:

- ◯ A
- ◯ B
- ◯ C
- ◯ D

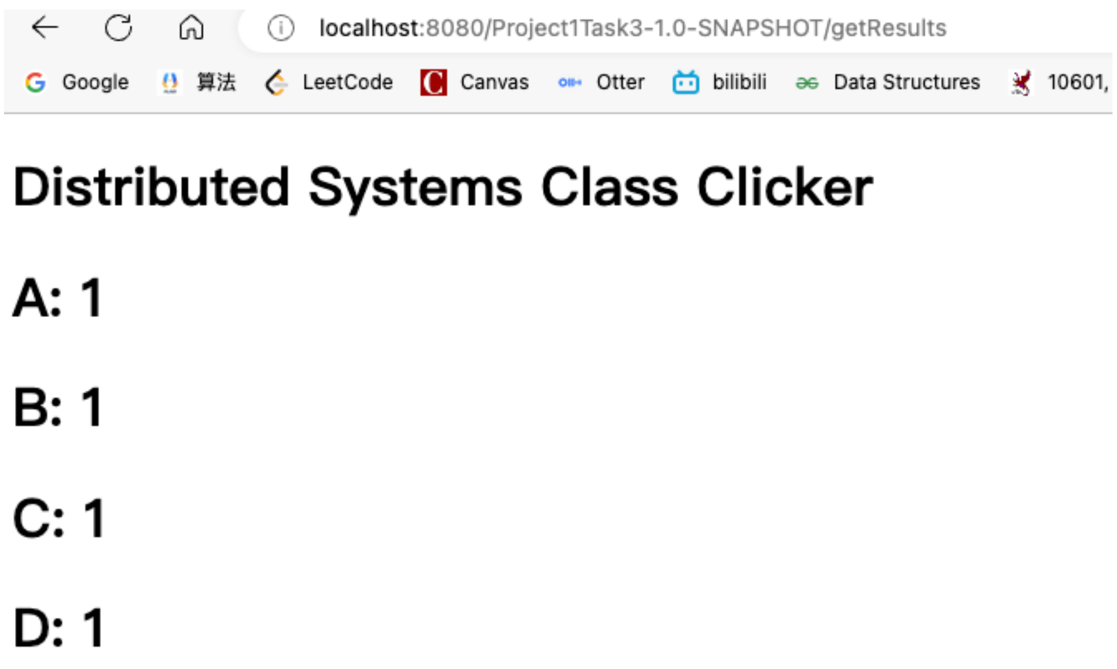

localhost:8080/Project1Task3-1.0-SNAPSHOT/getP1T3Servlet

Distributed Systems Class Clicker

Your "A" has been registered

Submit your answer to the current question:

- ◯ A
- ◯ B
- ◯ C
- ◯ D



Distributed Systems Class Clicker

A: 1

B: 1

C: 1

D: 1

Code Snippet

Java code that produces the output page and the results page.

•

```
• private void processRequest(HttpServletRequest request,
  HttpServletResponse response) throws ServletException,
  IOException {
      // Check if the request is from a mobile device
      setDoctypeAttribute(request);
      // Get the servlet path
      String path = request.getServletPath();
      String nextView;

      // Check the path and process the request accordingly
      if (path.equals("/getResults")) {
          nextView = processResults(request);
      } else {
          nextView = processAnswer(request);
      }
      // Get the RequestDispatcher for the next view
      RequestDispatcher view =
  request.getRequestDispatcher(nextView);
      // Forward the request to the next view
      view.forward(request, response);
```

```
    }

    // Processes the answer submitted by the user
    private String processAnswer(HttpServletRequest request) {
        // Get the answer from the request
        String answer = request.getParameter("answer");
        if (answer != null) {
            // Add the answer to the model
            answerClickerModel.addAnswer(answer);
            // Set the answer as a request attribute
            request.setAttribute("answer", answer);
            // Return the result view
            return "result.jsp";
        }
        // Return the prompt view if no answer is submitted
        return "input.jsp";
    }
```